IN THE CLAIMS:

1. (Currently Amended) A system adapted to distribute redundant information across disks of an array, the system comprising:

a storage operating system configured to invoke storage operations executed by a storage system, the storage operating system further configured to manage storage of information, including the redundant information and data, on blocks of the disks in response to disk access operations, the storage operating system including a storage module adapted to compute the redundant information in response to ~~placement~~ a layout of the data in stripes across the disks, the storage operating system maintaining at least one unallocated block per stripe for use by the storage module to store the computed redundant information, wherein the at least one unallocated block used to store the redundant information is located in any disk and wherein the location of the at least one unallocated block used to store the redundant information is dynamically allocated in a non-fixed pattern by the storage module before each write request is completed for each stripe.

2. (Original) The system of Claim 1 wherein the storage module is a disk array controller configured to compute the redundant information and reconstruct blocks lost due to failure of one or more of the disks.

3. (Original) The system of Claim 1 wherein the storage module is a RAID system configured to compute the redundant information and reconstruct blocks lost due to failure of one or more of the disks.

4. (Original) The system of Claim 3 wherein the storage operating system is further configured to implement a high-level module that maintains information about locations of the data on the disks.

5. (Original) The system of Claim 4 wherein the high-level module is a file system or database adapted to control layout of the data on the disks.

6. (Original)The system of Claim 5 wherein the storage operating system integrates the file system or database with the RAID system.

7. (Original)The system of Claim 6 wherein the file system or database is configured to determine block locations of the data on the disks and the RAID system is configured to determine the block locations of the redundant information on the disks.

8. (Original)The system of Claim 6 wherein the file system or database is configured to determine block locations of the data and the redundant information on the disks.

9. (Original) The system of Claim 8 wherein the file system or database renders balancing decisions to determine the block locations of the data and the redundant information on the disks.

10. (Original) The system of Claim 9 wherein the balancing decisions comprises one of different sizes of disks, different speeds of disks, and whether a disk is more heavily utilized than other disks.

11. (Original) The system of Claim 8 further comprising block allocation map structures used by the file system to determine the block locations of the data and the redundant information on the disks.

12. (Original)The system of Claim 11 wherein the redundant information is parity.

13. (Cancelled)

14. (Original) The system of Claim 1 wherein the storage module selects the at least one unallocated block to store the redundant information and wherein the storage module computes the redundant information using a redundant storage algorithm.

15. (Original) The system of Claim 14 wherein the selection of the at least one unallocated block to store redundant information is independent of the redundant storage algorithm.

16. (Previously Presented) The system of Claim 15 wherein the redundant storage algorithm is a symmetric algorithm.

17. (Original) The system of Claim 16 wherein the redundant information is parity.

18. (Original)The system of Claim 14 wherein the at least one unallocated block used to store the redundant information comprises two or more unallocated blocks used to store the redundant information.

19. (Original) The system of Claim 18 wherein the selection of the unallocated blocks to store redundant information is independent of the redundant storage algorithm used to compute the redundant information.

20. (Original) The system of Claim 19 wherein the redundant storage algorithm depends on positions of the blocks in the array.

21. (Previously Presented) The system of Claim 20 wherein the redundant storage algorithm is a symmetric and wherein the redundant information is parity.

22. (Currently Amended) A method for distributing redundant information across disks of an array, ~~the method~~ comprising ~~the steps of~~:

dividing each disk into blocks, the blocks being organized into stripes such that each stripe contains one block from each disk;

selecting any blocks in ~~the~~ a stripe layout of the stripe not used to contain data to contain the redundant information, wherein the block used to contain the redundant information is located in any disk and wherein the location of the block used to contain the redundant information is dynamically allocated in a non-fixed pattern before each write request is completed for each stripe; ~~and~~

computing the redundant information based on contents of all other blocks in the stripe layout, regardless of whether the blocks contain data; and

writing the stripe layout to the disks of the array.

23. (Original)The method of Claim 22 further comprising the step of determining which block in a stripe contains redundant information each time there is a write request to the stripe.

24. (Original) The method of Claim 23 further comprising the step of assigning a block to contain redundant information when each stripe is written.

25. (Original) The method of Claim 24 wherein the step of determining is performed by a high-level module of a storage system and wherein the steps of computing and assigning are performed by a storage module of the storage system.

26. (Currently Amended) The method of Claim 25 further comprising ~~the steps of~~:

maintaining, by the high-level module, at least one unallocated block per stripe for use by the storage module; and

providing an indication from the high-level module to the storage module of the unallocated block to contain parity.

27. (Original) The method of Claim 26 further comprising the step of reconstructing, using the storage module, a block that is lost due to failure of a disk.

28. (Original) The method of Claim 25 wherein the high-level module is a file system and wherein the storage module is one of an array controller and a RAID system.

29. (Currently Amended) The method of Claim 28 wherein the step of computing comprises ~~the step of~~ computing the redundant information in response to ~~placement~~ the stripe layout of the data in stripes across the disks.

30. (Currently Amended) The method of Claim 29 wherein the step of computing further comprises ~~the step of~~ computing the redundant information using algebraic and algorithmic calculations in response to the placement of the data on the array.

31. (Currently Amended) Apparatus for distributing redundant information across disks of an array, the apparatus comprising:

means for dividing each disk into stripes, with each stripe containing one block from each disk;

means for selecting any blocks in ~~the~~ a stripe layout of the stripe not used to contain data to contain redundant information, wherein the block used to contain the redundant information is located in any disk and wherein the location of the block used to contain the redundant information is dynamically allocated in a non-fixed pattern before each write request is completed for each stripe; ~~and~~

means for computing the redundant information based on contents of all other blocks in the stripe layout, regardless of whether the blocks contain data and

means for writing the stripe layout to the disks of the array.

32. (Original) The apparatus of Claim 31 further comprising means for determining which block or blocks in a stripe holds redundant information each time there is a write operation to the stripe.

33. (Currently Amended) A computer readable medium containing executable program instructions for distributing parity across disks of an array, the executable instructions comprising one or more program instructions for:

dividing each disk into stripes, with each stripe containing one block from each disk;

selecting any blocks in ~~the~~ a stripe layout of the stripe not used to contain data to contain parity, wherein the block used to contain the parity is located in any disk and wherein the location of the block used to contain the parity is dynamically allocated in a non-fixed pattern before each write request is completed for each stripe; ~~and~~

computing the parity based on contents of all other blocks in the stripe, regardless of whether the blocks contain data and

writing the stripe layout to the disks of the array.

34. (Previously Presented) The system of Claim 15 wherein the redundant storage algorithm is an asymmetric algorithm

35. (Previously Presented) The system of Claim 20 wherein the redundant storage algorithm is an asymmetric algorithm and wherein the redundant information is parity.

36. (Currently Amended) A method for distributing redundant information across disks of an array with a plurality of blocks on each disk, comprising:

determining which blocks are unallocated in a stripe across the disks;

reserving unallocated blocks for storing the redundant information in one or more reserved unallocated blocks;

arranging data in the stripe for the data to be stored in one or more allocated blocks across the disks of the array;

assigning the redundant information in a non-fixed pattern to the one or more reserved unallocated blocks; and

writing the data in the allocated blocks and the redundant information in the one or more reserved unallocated blocks as the stripe across the disks of the array.

37. (Previously Presented) The method of claim 36, further comprising:

storing parity information as the redundant information in the one or more reserved unallocated blocks.

38. (Previously Presented) The method of claim 36, further comprising:

adding a disk to the array;

storing a second stripe across the array by determining one or more unallocated blocks across the array including the added disk, and writing the data to allocated blocks and the redundant information to the one or more unallocated blocks of the second stripe.

39. (Currently Amended) An apparatus for distributing redundant information across disks of an array with a plurality of blocks on each disk, comprising:

means for determining which blocks are unallocated in a stripe across the disks;

means for reserving unallocated blocks for storing the redundant information in one or more reserved unallocated blocks;

means for arranging data in the stripe for the data to be stored in one or more allocated blocks across the disks of the array;

means for assigning the redundant information in a non-fixed pattern to the one or more reserved unallocated block; and

means for writing the data in the allocated blocks and the redundant information in the one or more reserved unallocated blocks as the stripe across the disks of the array.

40. (Previously Presented) The apparatus of claim 39, further comprising:

means for storing parity information as the redundant information in the one or more reserved unallocated blocks.

41. (Previously Presented) The apparatus of claim 39, further comprising:

means for adding a disk to the array;

means for storing a second stripe across the array by determining one or more unallocated blocks across the array including the added disk, and writing the data to allocated blocks and the redundant information to the one or more unallocated blocks of the second stripe.

42. (Currently Amended) A system for distributing redundant information across disks of an array with a plurality of blocks on each disk, comprising:

a storage operating system configured to invoke storage operations executed by a storage system, the storage operating system further configured (i) to determine which blocks are unallocated in a stripe across the disks, (ii) to reserve unallocated blocks for storing the redundant information in one or more reserved unallocated blocks, (iii) to arrange data in the stripe for the data to be stored in one or more allocated blocks across the disks of the array, (iv) to assign the redundant information in a non-fixed pattern to the one or more reserved unallocated block, and (iv) to write the data in the allocated blocks and the redundant information in the one or more reserved unallocated blocks as the stripe across the disks of the array.

43. (Previously Presented) The system of claim 42, wherein the redundant information is parity information.

44. (Currently Amended) A computer readable medium containing executable program instructions for distributing parity across disks of an array, the executable instructions comprising one or more program instructions for:

determining which blocks are unallocated in a stripe across the disks, where each disk has a plurality of blocks;

reserving unallocated blocks for storing the redundant information in one or more reserved unallocated blocks;

arranging data in the stripe for the data to be stored in one or more allocated blocks across the disks of the array; and

assigning the redundant information in a non-fixed pattern to the one or more reserved unallocated block; and

writing the data in the allocated blocks and the redundant information in the one or more reserved unallocated blocks as the stripe across the disks of the array.

Please add new claims 45 *et al.*

45. (New) The system of claim 1, wherein the non-fixed pattern is created by the redundant information being stored in any block remaining after the data is allocated to blocks of the stripe.

46. (New) A method for distributing redundant information across disks of an array with a plurality of blocks on each disk, comprising:

allocating data to blocks in a stripe layout, where the stripe layout represents a stripe across the disks of the array;

reserving unallocated blocks for storing the redundant information in one or more reserved unallocated blocks;

assigning the redundant information to the one or more reserved unallocated blocks; and

writing the data in the allocated blocks and the redundant information in the one or more reserved unallocated blocks as the stripe across the disks of the array.

47. (New) The method of claim 46, further comprising:

storing parity information as the redundant information in the one or more reserved unallocated blocks.

48. (New) The method of claim 46, further comprising:

adding a disk to the array;

storing a second stripe across the array by determining one or more unallocated blocks across the array including the added disk, and writing the data to allocated blocks and the redundant information to the one or more unallocated blocks of the second stripe.

49. (New) The method of claim 46, further comprising:

computing the redundant information based on contents of other blocks in the stripe layout, regardless of whether the blocks contain data or are empty.

50. (New) The method of claim 49, wherein the step of computing uses a symmetric algorithm and wherein the redundant information is parity.

51. (New) The method of claim 49, wherein the step of computing uses an asymmetric algorithm and wherein the redundant information is parity.

52. (New) The method of claim 46, wherein the step of assigning the redundant information is assigned in a non-fixed pattern.

53. (New) The method of claim 52, wherein the non-fixed pattern is created by the redundant information being stored in any block remaining after the data is allocated to blocks of the stripe.